

# System and Method for Image Enhancement, Dynamic Range Compensation, and Illumination Correction

## Technical Field

The technical field is enhancement of digital images.

## Background

The human visual system is typically able to adapt to lighting variations across scenes, perceiving details in regions with very different dynamic ranges. Most image recording systems, however, fail this dynamic range compression task. As a result, images produced by these image recording systems are often of poor quality, compared to images produced by human perception. Another task that is often poorly performed by the image recording systems is that of color constancy. Humans perceive color in a way that is fairly independent of scene illumination, whereas the image recording systems are strongly influenced by spectral shifts.

The above problems can be stated mathematically by describing a relationship between an acquired image  $S$ , a reflectance of objects of the image  $R$ , and an illumination  $L$  in a pixel-wise multiplication, or:

$$S = L \cdot R.$$

This expression means that at each point in the image  $S$ , the color value is the multiplication of the reflectance value by the illumination value. Given an image  $S$ , the problem to be solved is removal of the effects of illumination and recovery of the reflectance image. That is, given  $S$ , find both  $R$  and  $L$ . However, calculation of both  $R$  and  $L$  is typically not possible. The solution then, is to generate a methodology that can estimate  $R$  and  $L$ .

Retinex theory deals with compensation for illumination in images. The goal of the Retinex theory is to decompose a given image  $S$  into the reflectance image  $R$ , and the illumination image  $L$ , such that at each point  $(x,y)$  in the image domain,  $S(x,y)$  equals the product of  $R(x,y)$  and  $L(x,y)$ . The benefits of such decomposition include the possibility of removing illumination effects of back/front lighting, enhancing photographs and other image capture methods that include spatially varying illumination, such as images that contain indoor and outdoor zones, and correcting colors in images by removing illumination-induced color shifts.

As noted above, recovering the illumination  $L$  from a given image  $S$  is known to be a mathematically ill-posed problem, and known algorithms vary in the manner and effectiveness of overcoming this limitation. The Retinex approach provides the framework for one such method. The Retinex methodology was motivated by Edward Land's research

1 of the human visual system, which is described in R.H. Land, "The Retinex Theory of Color  
2 Vision," *Sci. Amer.*, Vol. 237, pp. 108-128 (1977).

3 The first Retinex algorithms were of the random walk type. Subsequent Retinex  
4 algorithms used homomorphic filtering. Yet another group of Retinex algorithms are based  
5 on solving a Poisson equation.

6 Figure 1 is a block diagram of an algorithm 10 that is representative of the general  
7 class of prior-art Retinex algorithms. In Figure 1, an input image  $S$  11 is applied to a  
8 logarithmic module 12 to produce a logarithmic version 13 of the input image  $S$  and its  
9 two components, illumination  $L$  and reflectance  $R$ . That is,  $s = \log S$ ,  $l = \log L$ , and  $r = \log R$ ,  
10 and thereby,  $s = l + r$ . Using the logarithmic version 13 of the input images, an estimator  
11 module 14 computes an estimate 17 of the illumination, designated in Figure 1 as  $l^*$ . The  
12 estimate 17 ( $l^*$ ) is then combined with the logarithmic version 13 ( $s$ ) at summer 18 to  
13 produce an estimate 19 of the reflectance (designated  $r^*$ ). Finally, the estimate 19 ( $r^*$ ) is  
14 converted from a logarithm to a number (antilogarithm) corresponding to the logarithm at  
15 expander 20, to produce a real number value as an estimate 21 of the reflectance (designated  
16 as  $R^*$ ). Prior art Retinex algorithms usually employ the same process as shown in Figure 1.

17 The Retinex-based algorithms take several different forms. One such form is the  
18 random walk algorithm, which is a discrete time random process in which the "next pixel  
19 position" is chosen randomly from neighbors of the current pixel position. Random walk  
20 type Retinex algorithms are variants of the following basic formulation: A large number of  
21 walkers are initiated at random locations of the logarithmic version 13 ( $s$ ), adopting a gray-  
22 value of their initial position. An accumulator image  $A$  that has the same size as  $s$  is  
23 initialized to zero. As a walker walks around, the walker updates  $A$  by adding the value of  
24 the walker to each position  $(x,y)$  that the walker visits. The illumination image is obtained by  
25 normalizing the accumulator image  $A$ , i.e., the value at each position of the accumulator  
26 image  $A$  is divided by the number of walkers that visited that position.

27 By using many walkers with long paths, one can easily verify that the accumulator  
28 value at each position converges to a Gaussian average of that position's neighbors.

29 Another type of Retinex algorithm uses homomorphic filtering, where a low-pass  
30 filter is used to reconstruct  $l$  from  $s$ . Homomorphic Retinex algorithms are based on the fact  
31 that the reflectance image  $R$  corresponds to the sharp details in the image (i.e., the edges),  
32 whereas the illumination  $L$  is expected to be spatially smooth. Then, a reasonable guess for  $l$   
33 is  $l^* = \text{LP}\{s\}$ , where LP is a convolution with a wide Gaussian kernel. Thus, the Retinex

algorithm using homomorphic filtering actually applies the same process as the random walk algorithms by a single direct convolution.

Since the illumination  $L$  is expected to be spatially smooth, the derivative of the illumination  $L$  should be close to zero. However, considering the assumption that the reflectance  $R$  is piece-wise constant, the derivative of the reflectance  $R$  should vanish almost everywhere, with high values along edges of an image. Taking the derivative of the sum  $s = l + r$  and clipping out high derivative peaks, implies that the clipped derivative signal corresponds only to the illumination  $L$ . Poisson equation-type Retinex algorithms that rely on the Mondrian world model, use the above assumptions on the reflectance  $R$  as a piece-wise constant image. Applying the Laplacian, and the following clipping operation:

$$\begin{aligned}\tau(\Delta s) &= \Delta s \quad \text{where } |\Delta s| < \tau \\ \tau(\Delta s) &= 0 \quad \text{otherwise,}\end{aligned}$$

yields the following Poisson equation

$$\Delta l^* = \tau(\Delta s)$$

A solution to the Poisson equation may involve an iterative procedure that effectively inverts the Laplacian operator. Improvements to the method involve extracting discontinuities from the image gradient magnitude instead of the Laplacian to provide better boundary conditions.

One solution involves use of an iterative algorithm having a “reset” non-linearity that enforces the constraint  $l \geq s$ . The algorithm performs the interactive procedure.

$$l_{n+1}^* = \max \left\{ \frac{l_n^* + s}{2}, \frac{l_n^* + D_n[l_n^*]}{2} \right\}$$

where  $D_n$  is a translation operator, shifting the image  $s$  by the  $n^{\text{th}}$  element of a sequence of spirally decaying translation vectors. Removing the max operation yields a simplified version

$$l_{n+1}^* = \frac{l_n^* + D_n[l_n^*]}{2}$$

The above equation is a simple averaging operation that smoothes the images. The non-linear (max) operation forces the illumination image  $l$  to satisfy the constraint  $l^* \geq s$ .

Despite use of these algorithms, current image recoding and image enhancement systems and methods cannot produce images that are of sufficient quality to be comparable to images as perceived by the human vision system. Thus, an improved method and system are required to better remove illumination effects and to recover the reflectance image.

## Summary

A system and a method solve the estimation problem of finding reflectance  $R$  and illumination  $L$ . The system and method use a functional of the unknown illumination  $L$  such that a minimum of the functional is assumed to yield a good estimate of the illumination  $L$ . Having a good estimate of the illumination  $L$  implies a good estimate of the reflectance  $R$ .

The functional uses a variational framework to express requirements for the optimal solution. The requirements include: 1) that the illumination  $L$  is spatially smooth; 2) that the reflectance values are in the interval  $[0,1]$  - thus, when decomposing the image  $S$ , the solution should satisfy the constraint  $L > S$ ; 3) that among all possible solutions, the estimate of the illumination  $L$  should be as close as possible to the image  $S$ , so that the contrast of the obtained  $R$  is maximal; and 4) that the reflectance  $R$  complies with typical natural image behavior (e.g., the reflectance is piece-wise smooth).

The four requirements lead to a well-behaved optimization problem having quadratic programming, which is convex, with a single solution. The method and the system use a numerical algorithm for the solution, where the algorithm is based on a multi-resolution decomposition of the image  $S$ .

The method and the system are adaptable to different image situations. In a first situation, given a color image in the RGB color-space, applying the algorithm on the three color channels separately produces color correction. An alternative embodiment involves converting the input image  $S$  into a chromatic-luminance color space. The algorithm is then applied to the luminance alone, resulting in an image that preserves the input colors, and improves the local contrast.

A second situation accounts for the fact that although the illumination  $L$  is assumed to be spatially smooth on a global level, the assumption is sometimes wrong. This situation is shown dramatically in Figure 9, where the illumination  $L$  is expected to have a sharp edge on the window boundaries. In situations such as that shown by the example of Figure 9, allowing the illumination  $L$  to include sharp edges may be enabled by forcing piece-wise smoothness on the illumination  $L$ , rather than assuming global smoothness.

A third situation does not use the assumption of piece-wise smoothness for the reflectance  $R$ . Instead, a better quality prior image  $S$  is used. The better prior  $S$  allows use of multi-resolution and anisotropic considerations.

A fourth and final situation accounts for the fact that some illumination effect is needed in order to give the human user a natural feel of the recorded image. Thus, the

1 method and the system may include a process that partially returns the illumination such that  
2 a final output image includes some illumination, but with improved contrast.

3 Using the four requirements enumerated above, the variational framework and the  
4 algorithm provide a novel solution to the problem of estimating illumination. The variational  
5 framework and the algorithm may be applied in many image recording situations, including  
6 in digital cameras as an automatic illumination compensation module, aimed at improving  
7 image reproduction quality, and in scanners and printers, as a special effect that improves the  
8 visual quality of a scanned/printed image.

### 9 **Description of the Drawings**

10 The detailed description will refer to the following drawings, in which like numbers  
11 refer to like objects, and in which:

12 Figure 1 is a diagram of a prior art algorithm for estimating illumination;

13 Figure 2 is a block diagram of an improved algorithm for decomposing an image;

14 Figure 3 is a diagram of a system that uses the improved algorithm of Figure 2;

15 Figure 4 is a flowchart showing a process according to the algorithm of Figure 2;

16 Figure 5 is a flowchart of a sub-routine of the process of Figure 4;

17 Figure 6 is a flowchart of a further sub-routine of Figure 4;

18 Figure 7 is a block diagram of an alternate improved algorithm;

19 Figure 8 is a block diagram of yet another alternate improved algorithm; and

20 Figure 9 is an image triplet showing the effects of illumination on an image.

### 21 **Detailed Description**

22 The human visual system is typically able to adapt to lighting variations across scenes  
23 (for example, shadows, strong illumination source), perceiving details in regions with very  
24 different dynamic ranges. Most image recording systems (e.g., cameras), however, fail this  
25 dynamic range compression task. As a result, images produced by these image recording  
26 systems are often of poor quality, compared to images produced by human perception.

27 Another task that is often poorly performed by the image recording systems is that of color  
28 constancy. Humans perceive color in a way that is fairly independent of scene illumination,  
29 whereas the images recording systems are strongly influenced by spectral shifts.

30 The above problems can be mathematically formulated by describing a relationship  
31 between an acquired image  $S$ , a reflectance  $R$  of objects of the image, and an illumination  $L$   
32 in a pixel-wise multiplication, or:

$$33 \quad S = L \cdot R.$$

1 This expression means that at each point in the image  $S$ , the color value is the multiplication  
2 of the reflectance value by the illumination value. The effect of this relationship can be seen  
3 by a simple experiment. A person or other object in a room is placed near a window (or other  
4 light source) during daylight such that a strong light source (i.e., bright daylight) illuminates  
5 part of an image  $S$  to be recorded. Thus, the illumination is strong through the window and  
6 weak inside the room. A photograph is then taken, and the recorded image  $S$  is decomposed  
7 into reflectance  $R$  and illumination  $L$ . The person can barely be seen in the image  $S$  and the  
8 illumination  $L$ , but can be seen in the reflectance  $R$ .

9 Given an image  $S$ , the problem to be solved is removal of the effects of illumination  
10 and recovery of the reflectance image. That is, given  $S$ , find both  $R$  and  $L$ . However,  
11 calculation of either  $R$  or  $L$  is typically not possible. The solution then, is to generate a  
12 methodology that can estimate  $R$  and  $L$ .

13 Figure 2 is a block diagram of an improved algorithm 100 that may be used to  
14 estimate the illumination  $L$  and the reflectance  $R$ . In Figure 2, an image  $S$  101 is input to a  
15 logarithm module 102 to produce a logarithm  $s$  103 of the image  $S$  101 and its two  
16 components, illumination  $L$  and reflectance  $R$ . That is,  $s = \log S$ ,  $l = \log L$ , and  $r = \log R$ , and  
17 thereby,  $s = l + r$ . Using the logarithm  $s$  103, an image processing module 104 uses a  
18 Projected Normal Steepest Descent or similar algorithm, with multi-resolution processing, to  
19 compute an estimate 107 of the illumination, designated in Figure 2 as  $l^*$ . The estimate 107  
20 ( $l^*$ ) is then combined with the output 103 ( $s$ ) at summer 108 to produce an estimate 109 of  
21 the reflectance (designated  $r^*$ ). Finally, the estimate 109 ( $r^*$ ) is converted from a logarithm  
22 to its corresponding base number value at exponential module 110, to produce a number  
23 value as an estimate 111 of the reflectance (designated as  $R^*$ ).

24 Figure 3 is a block diagram of a representative component (a digital camera) 112 that  
25 uses the system and the method described herein. The component 112 includes an image  
26 capture device 114 that captures an image. The image capture device may include lenses, a  
27 CCD array, and other sub-components. An image converter 115 may be an optional  
28 subcomponent used for pre-processing of the captured image. An image processor 116 may  
29 include the necessary processing software and algorithms to enhance the captured image  
30 according to the methods and systems described herein. The component 112 may include an  
31 image memory 117 that stores captured images, both before and after processing by the  
32 image processor 116. Finally, a display device 118, which may be a liquid crystal display  
33 device, for example, provides a visual display for a user of the component 112.

To arrive at the solution to the problem of estimating  $R$  and  $L$ , a method and a system for applying the method begins by formulating the estimation problem of finding  $R$  and  $L$  from  $S$  as an optimization problem. That is, a functional of one of the unknowns (either  $L$  or  $R$ ) is provided such that a minimum of the functional yields the desired result. The formulation uses a variational framework to express requirements from the optimal solution. The framework embodies the following requirements:

1. The illumination  $L$  is spatially smooth.
2. The reflectance  $R$  is restricted to the unit interval, which adds the constraint  $L \geq S$ . Since the log function is monotone,  $l \geq s$ .
3. Setting  $l = \text{Const}$ , where  $\text{Const}$  is any constant above the maximal value of  $s$ , yields a trivial solution that satisfies the two previous requirements. Thus, a requirement is added such that the illumination image  $l$  is close to the intensity image  $s$ , i.e., the value of  $l$  minimizes a penalty term of the form  $\text{dist}(l, s)$ , e.g., the  $L_2$  norm  $(l - s)^2$ .
4. The reflectance image  $s = l - r$  can be assumed to have a high prior probability. One of the simplest prior functions used for natural images assigns high probability to spatially smooth images.
5. The illumination continues smoothly as a constant beyond the image boundaries. This is an artificial assumption required for boundary conditions that would have minor effect on the final results.

Collecting all the above requirements into one expression yields the following penalty functional:

$$\begin{aligned} \text{Minimize: } F[l] &= \int_{\Omega} (|\nabla l|^2 + \alpha(l - s)^2 + \beta|\nabla(l - s)|^2) dx dy \\ \text{Subject to: } l &\geq s \quad \text{and} \quad \langle \nabla l, \vec{n} \rangle = 0 \quad \text{on } \partial\Omega, \end{aligned} \quad (1)$$

where  $\Omega$  is the support of the image,  $\partial\Omega$  is the image boundary, and  $\vec{n}$  is the normal to the boundary.  $\alpha$  and  $\beta$  are free non-negative real parameters. In the functional  $F[l]$ , the first penalty term  $(|\nabla l|^2)$  forces spatial smoothness on the illumination image. This choice of smoothness penalty is natural, keeping in mind that minimizing  $\int_{\Omega} (|\nabla l|^2) dx dy$  translates into the Euler-Lagrange (EL) equation  $\Delta l = 0$ . The steepest descent solution to the first penalty term is a Gaussian smoothing operation with increasing variance of the initial condition. As mentioned in the previous section, several authors proposed Gaussian smoothing of  $s$  for the illumination reconstruction.

The second penalty term  $(l - s)^2$  forces a proximity between  $l$  and  $s$ . The difference between these images is exactly  $r$ , which means that the norm of  $r$  should be small (i.e.,  $R$  tends to white). This term is weighted by the free parameter  $\alpha$ . The main objective of this term is a regularization of the problem that makes the second penalty term better conditioned. Notice that, in addition, the solution  $l$  is forced to be  $l \geq s$ .

The third penalty term is motivated by the Bayesian prior probability theory and encourages the reflectance image  $r$  to be a “visually pleasing image.” In practice, this prior probability expression forces  $r$  to be spatially smooth. The third penalty term is weighted by the free parameter  $\beta$ . More complicated Bayesian expressions may be used allowing sharp edges, textures,  $1/f$  behavior, etc. As long as this expression is purely quadratic, the above minimization problem remains fairly simple.

The above-defined problem has a Quadratic Programming (QP) form. The necessary and sufficient conditions for the minimization problem are obtained with the Euler-Lagrange equations:

$$\forall (x, y) \in \Omega \left\{ \begin{array}{l} \frac{\partial F[l]}{\partial l} = 0 = -\Delta l + \alpha(l - s) - \beta \Delta(l - s) \text{ and } l > s \\ \text{or} \\ l = s \end{array} \right. \quad (2)$$

The differential equation does not have to hold when  $l = s$ .

The minimization problem is QP with respect to the unknown image  $l$ . A Projected Normalized Steepest Descent (PNSD) algorithm, accelerated by a multi-resolution technique, is used to solve this minimization problem.

The PNSD algorithm requires the application of a Normalized Steepest Descent (NSD) iteration that minimizes the functional  $F[l]$ , followed by a projection onto the constraints. A NSD iteration has the format:

$$l_j = l_{j-1} - \mu_{NSD} G$$

where  $l_j$  and  $l_{j-1}$  are the illumination images at step  $j$  and  $j - 1$ , respectively,  $G$  is the gradient of  $F[l]$ , and  $\mu_{NSD}$  is the optimal line-search step size. For Equation (2), the gradient of  $F[l]$  is given by:

$$G = -\nabla l_{j-1} + (\alpha - \beta \Delta)(l_{j-1} - s),$$



1 and  $\mu_{NSD}$  is given by:

$$\mu_{NSD} = \frac{\int |G|^2}{\int (\alpha |G|^2 + (1 + \beta) |\nabla G|^2)}$$

2 Using integration by parts,  $\int |\nabla G|^2 = - \int G \Delta G$  up to the boundary conditions.

3 An alternative approach is the Steepest Descent (SD) algorithm, where  $\mu_{NSD}$  is  
4 replaced by a constant value  $\mu_{SD}$ , such that:

$$\mu_{SD} \in \left( 0, \frac{2}{\lambda_{\max} \{-(1 + \beta) \Delta + \alpha I\}} \right)$$

5 where  $\lambda_{\max}\{A\}$  refers to the greatest eigenvalue of the linear operator  $A$ . This alternative  
6 method saves computations at the expense of a slightly slower convergence.

7 Finally, projecting onto the constraint  $l \geq s$  is done by  $l_j = \max(l_j, s)$ . In practice,  $G$   
8 can be calculated by:

$$9 \quad G = G_A + \alpha (l_{j-1} - s) - \beta (G_A - G_B)$$

10 where

$$11 \quad G_A \triangleq \Delta l_{j-1},$$

$$12 \quad G_B \triangleq \Delta s_k$$

13 Similarly,  $\mu_{NSD}$  is given by:

$$\mu_{NSD} = \frac{\mu_A}{\alpha \mu_A + (1 + \beta) \mu_B},$$

14 where

$$\mu_A \triangleq \int |G|^2,$$

$$\mu_B \triangleq \int |\nabla|^2.$$

15 The Laplacian may be approximated by a linear convolution with the kernel  $k_{LAP}$

$$K_{LAP} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$

1 and the integrations are approximated by summations:

$$\begin{aligned} \iint |G|^2 &\approx \sum_n \sum_m G[n, m]^2 \\ \iint |\nabla G|^2 &= - \iint G \Delta G \\ &\approx - \sum_n \sum_m G[n, m] (G * K_{LAP})[n, m] \end{aligned}$$

2 where  $G[m, n] = G(m\Delta x, n\Delta y)$ . In order to accommodate the boundary conditions, as given  
3 in Equation (1), the above convolution is applied on an expanded version of the image  $G$ .  
4 The extension is done by replicating the first and last columns and rows. After convolution,  
5 the additional rows and columns are removed.

6 Although simple, the PNSD algorithm usually converges slowly. Instead of general  
7 acceleration schemes, the fact that the unknown image  $I$  is assumed to be smooth may be  
8 used. Specifically, the method applies a multi-resolution algorithm that starts by estimating a  
9 coarse resolution image  $I$ , expands the resolution image  $I$  by interpolation, and uses the result  
10 as an initialization for the next resolution layer. This way, only a few iterations at each  
11 resolution are required for convergence.

12 Summarizing the above, a block diagram of an algorithm 120 for the solution of  
13 Equation (1) is shown in Figure 4. The algorithm 120 begins with input block 121, where an  
14 input to the algorithm 120 is an image  $s$  of size  $[N, M]$ , and the two parameters  $\alpha$  and  $\beta$ .

15 In initialization block 123, a Gaussian pyramid of the image  $s$  is computed. The thus-  
16 constructed pyramid contains  $p$  resolution layers (from fine (1) to coarse ( $p$ )) with the current  
17 resolution layer ( $k$ ) set to  $p$ . In block 125,  $T_k$  iterations of the PNSD algorithm are applied to  
18 the  $k$ th resolution layer, until all resolution layers are checked, block 127. In block 129, the  
19 next resolution layer is updated. When all resolution layers are processed, the result is the  
20 final output of the algorithm 120.

21 Figure 5 is a block diagram showing the initialization routine 123 in more detail. In  
22 block 131, a counter is initialized with  $l = 1$ . In block 132, the Gaussian pyramid is  
23 constructed by smoothing the image with a kernel, such as the kernel  $k_{PYR}$ :

$$K_{PYR} = \begin{bmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{bmatrix}$$

1 In block 133, the image is decimated by, for example, a 2:1 ratio. The process is repeated  
 2 (block 134) and the counter is incremented (block 135)  $p$  times where for this section  $p < \lg_2$   
 3  $(\min(M, N))$ . This process produces a sequence of images  $\{s_k\}_{k=1}^p$  conventionally named the  
 4 "Gaussian Pyramid of  $s$ ." The image  $s_1$  is the original image  $s$ , and  $s_p$  is an image with the  
 5 coarsest resolution for the Gaussian pyramid.

6 In block 137, the process sets  $k = p$ , i.e., starts at the coarsest resolution layer  $k$ , and  
 7 sets the initial condition  $I_0 = \max \{s_p\}$ .

8 Figure 6 is a block diagram of the PNSD processing main routine 125. The routine  
 9 125 starts at block 141, where for the  $k^{th}$  resolution layer, the routine 125 calculates:

$$G_B = \Delta_k s_k$$

10 Where:  $\Delta_k \zeta = \zeta * k_{LAP} 2^{-2(k-1)}$ ; namely a convolution of  $\zeta$  with the kernel  $k_{LAP}$  as  
 11 specified above, normalized by a factor  $2^{(2(k-1))}$ .

12 Then, for  $j = 1, \dots, T_k$ :

13 In block 143, the routine 125 calculates the gradient:

$$\begin{aligned} G_A &= \Delta_k I_{j-1}, \\ G &= G_A + \alpha(I_{j-1} - s_k) - \beta(G_A - G_B) \end{aligned}$$

14 In block 145, the routine 125 calculates  $\mu_{NSD}$ :

$$\begin{aligned} \mu_A &= \langle G, G \rangle, \\ \mu_B &= -\langle G, \Delta_k G \rangle, \\ \mu_{NSD} &= \mu_A / (\alpha \mu_A + (1 + \beta) \mu_B) \end{aligned}$$

15 Where:  $\langle G, F \rangle = \sum_{n=1}^N \sum_{m=1}^M G[n, m] F[n, m]$

16 In block 147, the routine 125 completes the NSD iteration:

$$l_j = l_{j-1} - \mu_{NSD} \cdot G,$$

1 In block 149, the result is projected onto the constraints:

$$l_j = \max\{l_j, s_k\}$$

2 The loop processing solves the intermediate problem:

$$\begin{aligned} \text{Minimize: } F_k[l] &= \int_{\Omega_k} (|\nabla l|^2 + \alpha(l - s_k)^2 + \beta|\nabla(l - s_k)|^2) dx dy \\ \text{Subject to: } l &\geq s_k \quad \text{and } \langle \nabla l, \vec{n} \rangle = 0 \quad \text{on } \partial\Omega \end{aligned}$$

3 Returning to Figure 4, in block 127, if  $k > 1$ , the result  $l_{7k}$  is up scaled (2:1 ratio) by  
4 pixel replication into the new  $l_0$ , the initialization for the next resolution  $k - 1$  layer. The  
5 resolution layer is updated  $k = k - 1$ , and the algorithm proceeds by going again to block 125.  
6 If  $k = 1$ , the result  $l_{71}$ , is the final output of the algorithm 120.

7 By setting the parameters  $\alpha = \beta = 0$ , and removing the constraint  $l \geq s$ , the algorithm  
8 120 reduces to the Homomorphic filtering process that was shown to be equivalent to the  
9 basic formulation of the random walk algorithms.

10 Using the method and the system described above, the solution to Equation (1), and  
11 the convergence of the numerical algorithm 120 can be shown to be unique. That is, the  
12 *variational optimization problem P, given by*

$$\begin{aligned} \text{Minimize: } F_k[l] &= \int_{\Omega_k} (|\nabla l|^2 + \alpha(l - s_k)^2 + \beta|\nabla(l - s_k)|^2) dx dy \\ \text{Subject to: } l &\geq s_k \quad \text{and } \langle \nabla l, \vec{n} \rangle = 0 \quad \text{on } \partial\Omega \end{aligned}$$

13 *with  $\alpha \geq 0$  and  $\beta \geq 0$ , has a unique solution.*

14 The algorithm 120 has thus far been described with respect to a single channel. In  
15 another embodiment, the system and the method are applied to color images. When  
16 processing color images, one option is to deal with each color channel separately. Such  
17 channel-by-channel processing may be referred to as 'RGB Retinex' processing. Treating the  
18 R, G, and B channels separately usually yields a color correction effect. For example,  
19 applying RGB Retinex processing to a reddish image should modify the illumination in such  
20 a way that the red hue is removed and the resulting image is brightened and corrected.  
21 Therefore, for some image types, RGB Retinex processing improves the colors. For other

1 image types, such color correction may cause color artifacts that exaggerate color shifts, or  
2 reduce color saturation.

3 In yet another embodiment, colors are mapped into a luminance/chrominance color  
4 space, such as HSV. Next, the Retinex correction is applied, but only to the luminance or V  
5 layer, and then the colors are mapped back to the RGB domain. This method may be referred  
6 to as 'HSV Retinex'. Color shifts in such cases are less-likely. The advantage of HSV  
7 Retinex processing is that only a single channel is processed. However, using this  
8 embodiment colors are not corrected with respect to hue (H), and saturation ( $\bar{S}$ ).

9 The reflectance image  $R$  obtained by the Retinex process is sometimes an over-  
10 enhanced image. This can be explained by the facts that i) the human visual system usually  
11 prefers illumination in the image, and that ii) removal of all the illumination exposes noise  
12 that might exist in darker regions of the original image.

13 In still another embodiment, a corrected version of the reconstructed illumination is  
14 added back to the constructed reflectance image. Figure 7 describes this operation. In Figure  
15 7, an algorithm 200 computes the illumination image  $L = \exp(I)$  from the intensity image  $S =$   
16  $\exp(s)$ , and the reflectance image  $R = S/L$ . This is shown where an input image  $S$  201 is  
17 applied to a Retinex module 202 to produce an illumination image  $L$  203. The illumination  
18 image  $L$  203 is tuned by a Gamma correction operation 204 with a free parameter  $\gamma$ , to  
19 obtain a new illumination image  $L'$  209. A reflectance module 206 computes reflectance  $R$   
20 207 from the input image  $S$  201 and the illumination image  $L$  203. The new illumination  
21 image  $L'$  209 is multiplied by the reflectance  $R$  207 at 210 to produce an output image  $S'$  211,  
22 where  $S' = L' R$ . The Gamma correction is performed by

$$L' = W \cdot \left[ \frac{L}{W} \right]^{\frac{1}{\gamma}},$$

23 where  $W$  is the white value (equal to 255 in 8-bit images).

24 The final result  $S'$  is given, therefore, by:

$$\begin{aligned} S' &= L' \cdot R = \frac{L'}{L} S \\ &= W \frac{(L/W)^{1/\gamma}}{L} S = \frac{S}{(L/W)^{1-1/\gamma}} \end{aligned}$$

25 For  $\gamma = 1$ , the whole illumination is added back, and therefore  $S' = S$ . For  $\gamma = \infty$ , no  
26 illumination is returned, and  $S' = R W$ , which is the same reflectance image,  $R$ , as obtained

1 by the algorithm 100, stretched to the interval  $[0, W]$ . This later case can also be considered  
 2 as multiplying a maximum constant illumination  $W$  to the reflectance image  $R$ .

3 An analog to adding the illumination to the final image can be found in the  
 4 homomorphic filtering approach. A linear filter for the illumination calculation in the log  
 5 domain removes high-pass spatial components of  $s$ , yet also attenuates the low-pass  
 6 components by a factor of  $\gamma_i$  (where  $i$  stands for *illumination*). This is an analog to a gamma  
 7 correction of the illumination with  $\gamma = \gamma_i$ , since the equation for  $S'$  can be written in the form:

$$\frac{S'}{W} = \left( \frac{L}{W} \right)^{1/\gamma} \cdot R,$$

8 and therefore:

$$s' - w = \frac{1}{\gamma} (l - w) + r$$

$$= \frac{1}{\gamma} (\text{low-pass components}) + (\text{high-pass components}).$$

9 In an ideal environment, the illumination  $L$  is assumed to be piece-wise smooth. That  
 10 is,  $L$  is smooth everywhere except at strong discontinuities of the input image  $S$ . This ideal  
 11 can be approached in a number of ways. First, referring again to Figure (1), both the  
 12 illumination  $L$  and the reflectance  $R$  are required to be smooth. However, these requirements  
 13 contradict each other because  $l$  plus  $r$  must equal  $s$ . Thus  $l$  and  $r$  share the discontinuities.  
 14 The parameter  $\beta$  arbitrates this contradiction and makes sure that appropriate parts of each  
 15 discontinuity are apportioned to  $l$  and  $r$ .

16 A second approach ensures that the illumination  $l$  smoothness requirement be valid  
 17 everywhere except over discontinuities of  $s$ , and that the reflectance smoothness be valid over  
 18 discontinuities of  $s$ .

19 This second approach requires computing a weight function  $w(\nabla s)$  that obtains values  
 20 close to 1 over most parts of the image  $s$ , and close to 0 over strong discontinuities. Equation  
 21 (1) then becomes:

$$22 \quad F[l] = \int_{\Omega} \left( w(\nabla s) |\nabla l|^2 + \alpha |l - s|^2 \right) + \beta (1 - w(\nabla s)) |\nabla (l - s)|^2 dx dy \quad (3)$$

1 The projected steepest descent solution to Equation (3) composed of a series of gradient steps  
2 is:

$$3 \quad l_{i+1} = l_i + \mu [D_x^f(w(\nabla s) D_x^b(l_i)) + D_y^f(w(\nabla s) D_y^b(l_i)) - \alpha(l_i - s) + \\ \beta \cdot D_x^f((1 - w(\nabla s)) D_x^b(l_i - s)) + \beta \cdot D_y^f((1 - w(\nabla s)) D_y^b(l_i - s))] \\ 4$$

and projections:

$$5 \quad l_i = \max \{ l_i, s \}$$

6 Where:

7  $\mu$  is a constant

8  $D_x^f(.)$  is a forward derivative in the x-direction.

9  $D_x^b(.)$  is a backward derivative in the x-direction.

10  $D_y^f(.)$  is a forward derivative in the y-direction.

11  $D_y^b(.)$  is a backward derivative in the y-direction.

12 Figure 8 illustrates an algorithm 125a, which is an alternative to the algorithm 125 in  
13 algorithm 120, that may be used to solve the functional (Equation (3)).

14 In block 221, the algorithm 125a calculates  $w(\nabla s)$ ,  $\alpha_k$  and  $T_k$ :

$$15 \quad w(\nabla s_k) = \frac{1}{\left(1 + \frac{|\nabla s_k|^2}{Th}\right)^6}$$

16 where  $Th$  is a constant threshold,

$$17 \quad \text{and} \quad \alpha_k = \alpha_0 \cdot 2^{2k} \\ T_k = T_0 \cdot 2^{k+1}$$

18 Note that  $\alpha_k$  changes for each resolution level-a higher  $\alpha_k$  corresponds to a course resolution  
19 level.

20 In block 223, the algorithm 125a calculates:

$$21 \quad G = D_x^f(w(\nabla s) D_x^b(l_i)) + \\ 22 \quad D_y^f(w(\nabla s) D_y^b(l_i)) - \alpha(l_i - s) + \\ 23 \quad \beta D_x^f((1 - w(\nabla s)) D_x^b(l_i - s)) + \\ 24 \quad \beta D_y^f((1 - w(\nabla s)) D_y^b(l_i - s))$$

1 In block 225, the algorithm 125a completes the SD iteration:

2 
$$l_{i+1} = l_i + \mu \cdot G$$

3 In block 227, the result is projected into the constraints

4 
$$l_i = \max \{l_i, s_k\}.$$

5 In block 229, the algorithm 125a checks if  $i = T_k$ , and if so, processing according to  
6 the algorithm 220 ends.

HP: 10001200